

Version Control Guide



Concepts, Commands,
Workflows & Best Practices

VERSION CONTROL

A Version Control System (VCS) is software that helps developers track, manage, and collaborate on source code changes over time.

It maintains a complete history of modifications.

It is also known as software configuration management.

Types:

- Local vcs → keeps version on your local machines only
- Centralized vcs → one central server. It is also called client server architecture. It is supported by SVN/subversion, cvs.
- Distributed vcs → each user has a full copy. It is also called distributed architecture. It is supported by GIT tool, mercurial

Git Architecture

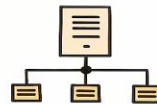
Local VCS

Versions kept on local machines only



Centralized VCS

One central server
Client server architecture



Distributed VCS

Each user has a full copy
Distributed architecture



□ 1. Git Basics

Git is a free and open-source distributed version control system to handle everything from small to every large project with speed and efficiency.

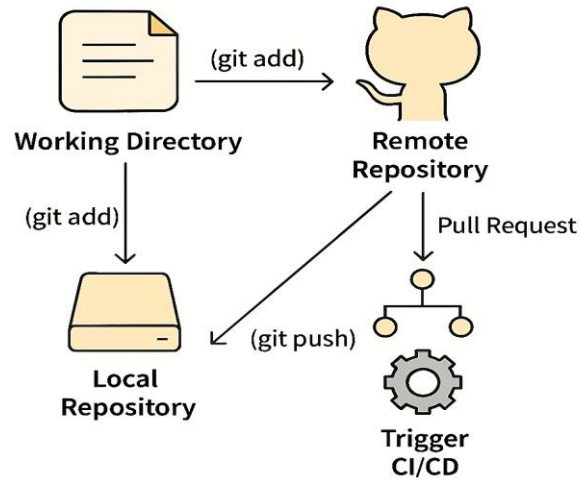
Features:

- Track versions of source code.
- Rollback to older versions.
- Auto-merge changes.
- Manage customer/release branches.

Common Commands:

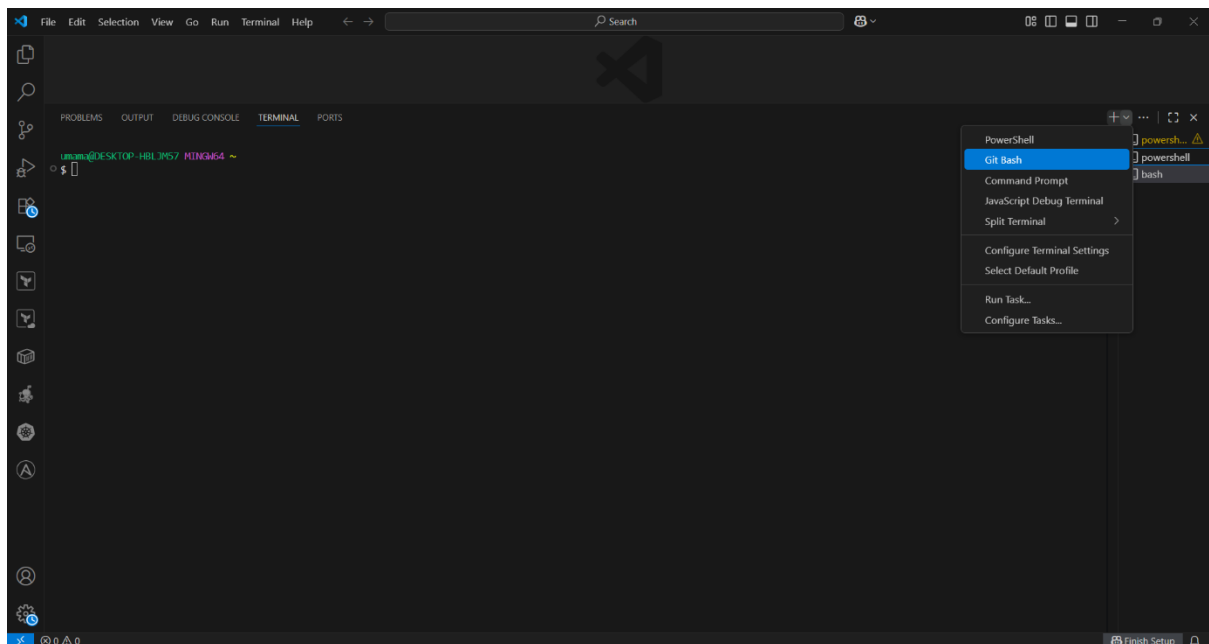
Command	Description	Example
git init	Initialize a new Git repository	git init myproject
git clone	Clone an existing repo	git clone <repo-url>
git status	Check repo status	git status
git add	Stage files for commit	git add file.txt
git commit	Save changes to repo	git commit -m "msg"
git log	View commit history	git log --oneline

Git local repo vs. remote repo (GitHub)

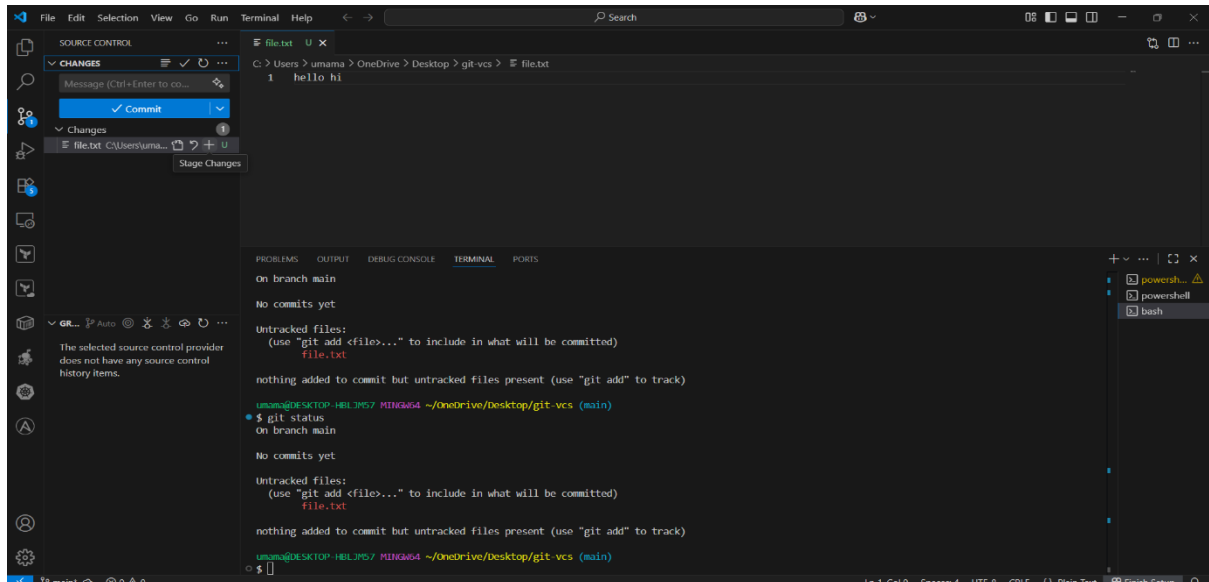


In vs code without commands, we push to github

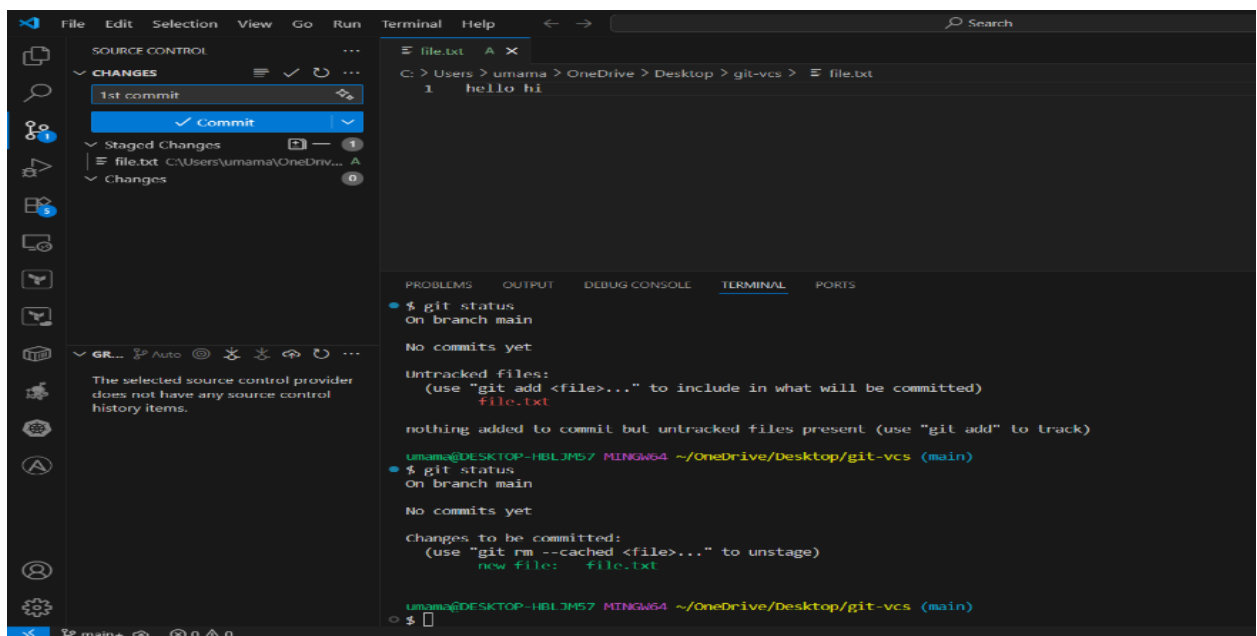
Create a folder on laptop open it with vscode click on terminal
→ new terminal → powershell to bash change



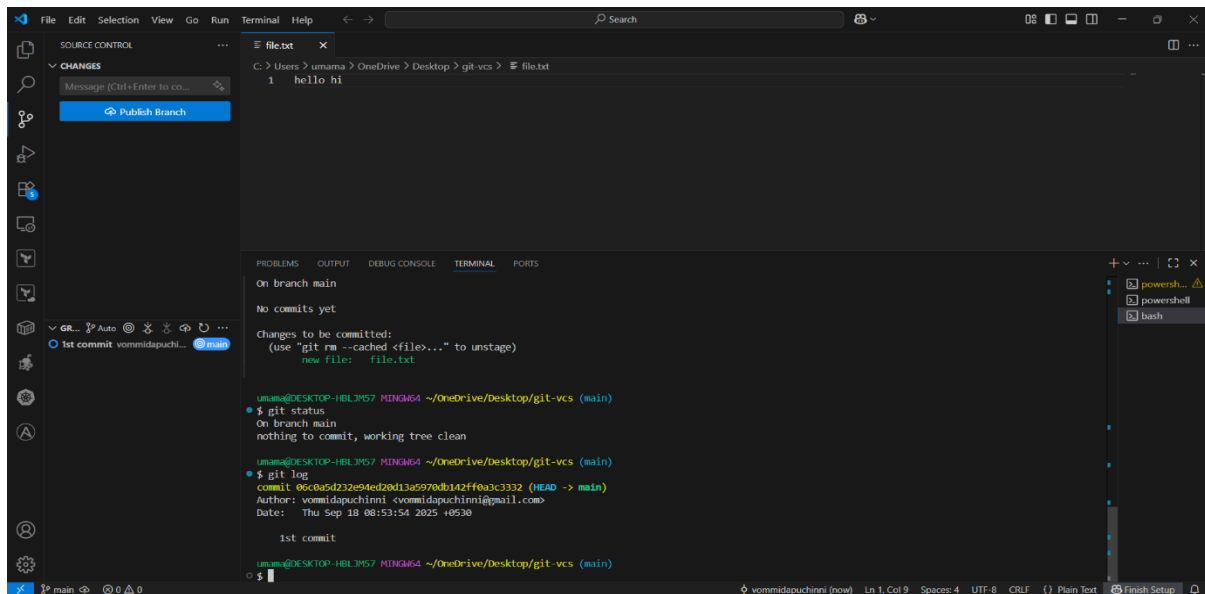
git init → create a file file.txt → enter content → save the file
click on source control on vscode left side → git status we see
in red colour → + click on it in left side it will save changes.



Give commit msg and click on commit



Check git status and git log we see 1st commit there



The screenshot shows the Visual Studio Code interface. The left sidebar has the 'Source Control' view open, showing a commit message 'hello hi' and a 'Publish Branch' button. The main editor area shows a file named 'file.txt' with the content 'hello hi'. The bottom panel shows the 'Terminal' view with the following output:

```
On branch main
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file.txt

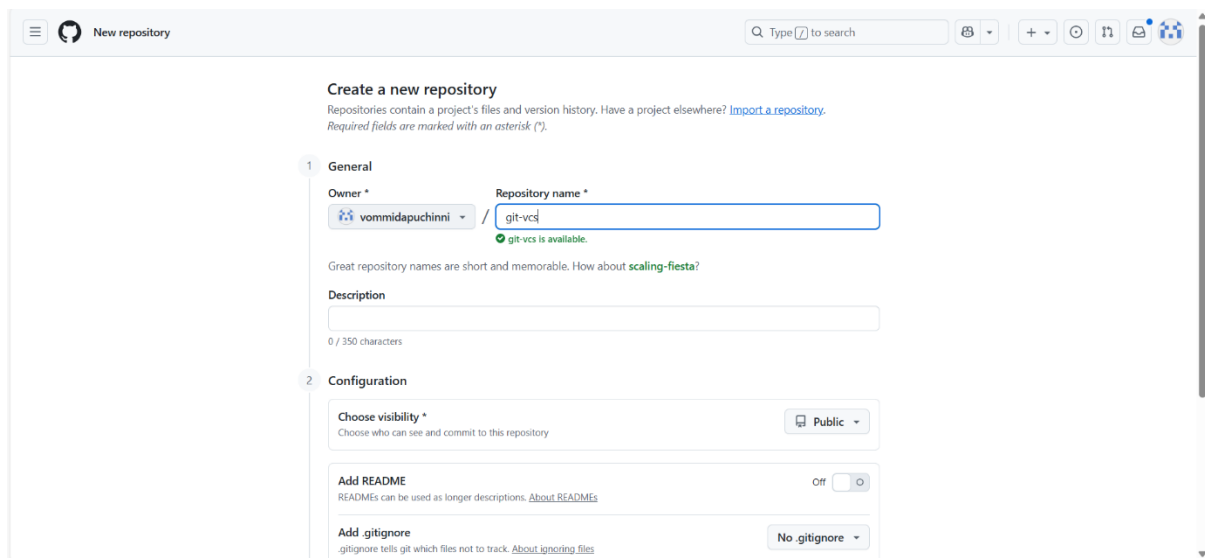
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git status
On branch main
nothing to commit, working tree clean

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit 06c8a5d232e94ed20d13a5970db142ff0a3c3332 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date:   Thu Sep 18 08:53:54 2025 +0530

    1st commit

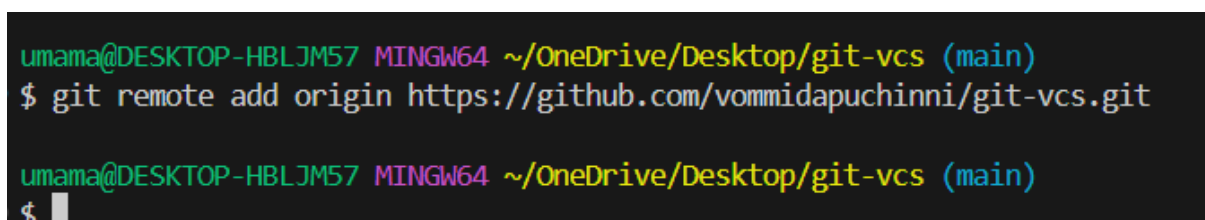
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$
```

Create remote repo with same folder name



The screenshot shows the GitHub 'New repository' page. The 'General' section is active, showing the 'Repository name' field with the value 'git-vcs' and a note that 'git-vcs is available'. The 'Configuration' section is also visible, showing 'Choose visibility' set to 'Public', 'Add README' set to 'Off', and 'Add .gitignore' set to 'No .gitignore'.

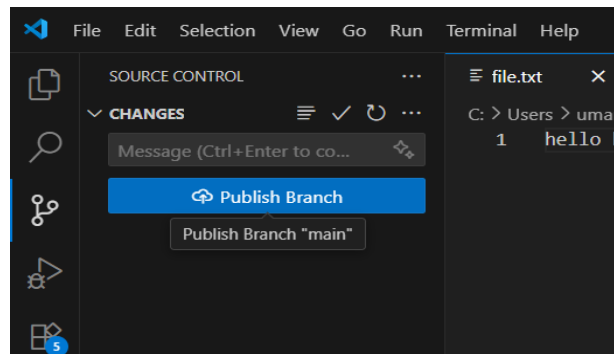
Paste url of remote



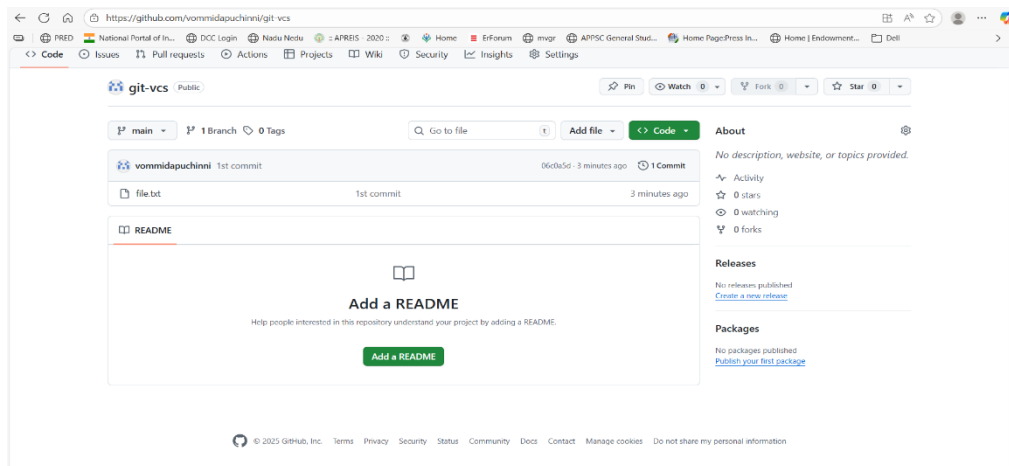
The screenshot shows a terminal window with the following commands and output:

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git remote add origin https://github.com/vommidapuchinni/git-vcs.git

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$
```



Click on publish branch. In git hub repo we see file is pushed



GitHub Repository Storage

Every repository uses Git to store data.

Git compresses and versions files — so each commit adds only changes, not full copies.

Key Notes:

- Soft limit per repository: **~1 GB**
- Hard limit per file: **100 MB**
- If you push a file >100 MB → GitHub rejects it
- Repos >1 GB trigger a warning
- Ideal repo size: **<500 MB**

2. Branching & Merging

Definition: Branches allow multiple lines of development in the same repository.

git branch → list the branches

git checkout -b dev → create a new branch and switch to it

data of existing branch is copied to new branch

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git branch
* main

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git checkout -b dev
Switched to a new branch 'dev'

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ ls
file.txt

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ vi file2

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ ls
file.txt  file2

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ ls
file.txt  file2

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$
```

When you commit that file then that data belongs to that particular branch


```

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ ls
file.txt file2

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git checkout dev
Switched to branch 'dev'

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git add .
warning: in the working copy of 'file2', LF will be replaced by CRLF the next time Git touches it

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git commit -m "2nd commit"
[dev 15d8399] 2nd commit
1 file changed, 1 insertion(+)
create mode 100644 file2

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ ls
file.txt file2

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ ls
file.txt

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$

```

We see 2 commits in dev branch and one commit in main branch

```

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git checkout dev
Switched to branch 'dev'

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git log
commit 15d8399f6ac519ad4a405d5fc731d7503ea8d0a6 (HEAD -> dev)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:00:36 2025 +0530

    2nd commit

commit 06c0a5d232e94ed20d13a5970db142ff0a3c3332 (origin/main, origin/HEAD, main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 08:53:54 2025 +0530

    1st commit

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit 06c0a5d232e94ed20d13a5970db142ff0a3c3332 (HEAD -> main, origin/main, origin/HEAD)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 08:53:54 2025 +0530

    1st commit

umama@DESKTOP-HBLJMS7 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$

```

Git merge: Merge another branch into current branch

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git merge dev
Updating 06c0a5d..15d8399
Fast-forward
 file2 | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 file2

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git branch
  dev
* main

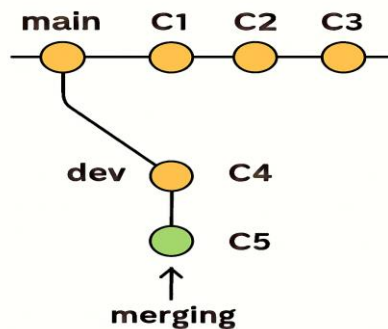
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ ls
file.txt  file2

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git branch -D dev
Deleted branch dev (was 15d8399).

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git branch
* main

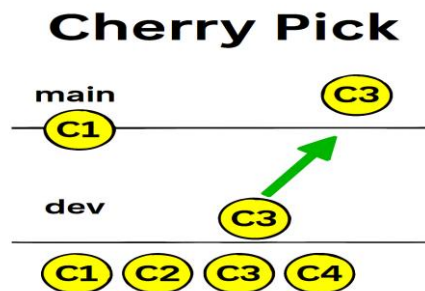
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$
```

Branching Tree



3. Cherry-pick: Apply a commit from one branch to another

We have 4 commits in dev branch and moving 3rd commit to the main branch



```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git log
commit 6ce27e0a95bc2eb27acfa70eb84356a155596aa1 (HEAD -> dev)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:33:11 2025 +0530

4th commit

commit d477de82ab7f423662c7a3bac3e2ca96a45479a2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:31:08 2025 +0530

3rd commit

commit 5df39a49fd252f2ad426dc483ffba2668134512b
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:30:14 2025 +0530

2nd commit

commit 06c0a5d232e94ed20d13a5970db142ff0a3c3332 (origin/main, origin/HEAD, main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 08:53:54 2025 +0530

1st commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git cherry-pick d477de82
[main c6d4b0f] 3rd commit
Date: Thu Sep 18 09:31:08 2025 +0530
1 file changed, 3 insertions(+)
create mode 100644 file3

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit c6d4b0ff9b13d935dea3441fb96dbd0898bf1d74 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:31:08 2025 +0530

3rd commit

commit 06c0a5d232e94ed20d13a5970db142ff0a3c3332 (origin/main, origin/HEAD)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 08:53:54 2025 +0530

1st commit
```

We see 3rd commit in main branch

4. Git tag: Assign a meaning full name with a specific version in the repo

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit c6d4b0ff9b13d935dea3441fb96dbd0898bf1d74 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:31:08 2025 +0530

    3rd commit

commit 06c0a5d232e94ed20d13a5970db142ff0a3c3332 (origin/main, origin/HEAD)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 08:53:54 2025 +0530

    1st commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git tag -a v1.0.0 -m "1st version" c6d4b0f

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git tag
v1.0.0

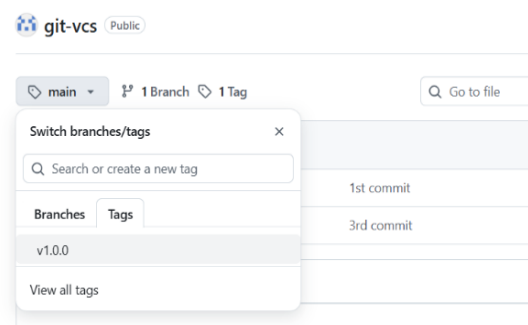
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$
```

When we push branch then tag will not be pushed separately, we have to push the tag


```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 279 bytes | 69.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/vommidapuchinni/git-vcs.git
    06c0a5d..c6d4b0f  main -> main
```

Like this git push origin tag version and We see the tag in git repo

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git push origin v1.0.0
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 162 bytes | 54.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/vommidapuchinni/git-vcs.git
 * [new tag]          v1.0.0 -> v1.0.0
```



Undoing Changes

 **5. Git reflog:** if we want to recover a deleted branch that was not merged or already merged by using the following command git reflog

Delete the branch → git branch -D branch name

If we want total data recovery then we use latest head

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git branch -D dev
Deleted branch dev (was 6ce27e0).

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git reflog
c6d4b0f (HEAD -> main, tag: v1.0.0, origin/main, origin/HEAD) HEAD@{0}: cherry-pick: 3rd commit
06c0a5d HEAD@{1}: checkout: moving from dev to main
6ce27e0 HEAD@{2}: commit: 4th commit
d477de8 HEAD@{3}: commit: 3rd commit
5df39a4 HEAD@{4}: commit: 2nd commit
06c0a5d HEAD@{5}: checkout: moving from main to dev
06c0a5d HEAD@{6}: reset: moving to 06c0a5d232e94ed20d13a5970db142ff0a3c3332
3821697 HEAD@{7}: commit: commit 3
15d8399 HEAD@{8}: merge dev: Fast-forward
06c0a5d HEAD@{9}: checkout: moving from dev to main
15d8399 HEAD@{10}: checkout: moving from main to dev
06c0a5d HEAD@{11}: checkout: moving from dev to main
15d8399 HEAD@{12}: commit: 2nd commit
06c0a5d HEAD@{13}: checkout: moving from main to dev
06c0a5d HEAD@{14}: checkout: moving from dev to main
06c0a5d HEAD@{15}: checkout: moving from main to dev
06c0a5d HEAD@{16}: commit (initial): 1st commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git checkout -b dev HEAD@{2}
Switched to a new branch 'dev'
```

We see all files and commits are recovered. For working together, we have to use collaborators in repo settings

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ ls
file1 file2 file3 file4

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$ git log
commit 6ce27e0a95bc2eb27acfa70eb84356a155596aa1 (HEAD -> dev)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:33:11 2025 +0530

    4th commit

commit d477de82ab7f423662c7a3bac3e2ca96a45479a2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:31:08 2025 +0530

    3rd commit

commit 5df39a49fd252f2ad426dc483ffba2668134512b
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:30:14 2025 +0530

    2nd commit

commit 06c0a5d232e94ed20d13a5970db142ff0a3c3332
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 08:53:54 2025 +0530

    1st commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (dev)
$
```

6. Git reset: default is mixed

- Create file and add content to it git add and commit it.
- Again, add some content and add and commit it.
- Again, add some data that also add and commit it

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs
$ git init
Initialized empty Git repository in C:/Users/umama/OneDrive/Desktop/git-vcs/.git/

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .
warning: in the working copy of 'f1', LF will be replaced by CRLF the next time Git touches it

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit -m "1st commit"
[main (root-commit) 34d3cd2] 1st commit
1 file changed, 3 insertions(+)
create mode 100644 f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .
warning: in the working copy of 'f1', LF will be replaced by CRLF the next time Git touches it

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit -m "2nd commit"
[main 3c04fbf] 2nd commit
1 file changed, 3 insertions(+)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .
warning: in the working copy of 'f1', LF will be replaced by CRLF the next time Git touches it

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit -m "3rd commit"
[main 818de3c] 3rd commit
1 file changed, 3 insertions(+)
```

Git log we see 3 commits

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit -m "3rd commit"
[main d321f6c] 3rd commit
1 file changed, 3 insertions(+)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit d321f6cf7f5e1fd647446f38dae7660656a2791d (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:05:16 2025 +0530

    3rd commit

commit 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:52:24 2025 +0530

    2nd commit

commit 34d3cd206a0c5515ce97731dd438aad027fdc2c2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:51:35 2025 +0530

    1st commit
```

git reset --soft: reset data from local repo → git reset --soft
<commit id>

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git reset --soft 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   f1
```

git reset --mixed: reset data from local repo, staging area. The mixed option is default.

git reset --mixed <commit id>

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git reset --mixed 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
Unstaged changes after reset:
M       f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   f1

no changes added to commit (use "git add" and/or "git commit -a")
```

git reset --hard: reset data from local repo, staging area and working dir also

git reset --hard <commit id>

we see data is removed

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git reset --hard 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
HEAD is now at 3c04fbf 2nd commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git log
commit 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date:   Thu Sep 18 09:52:24 2025 +0530

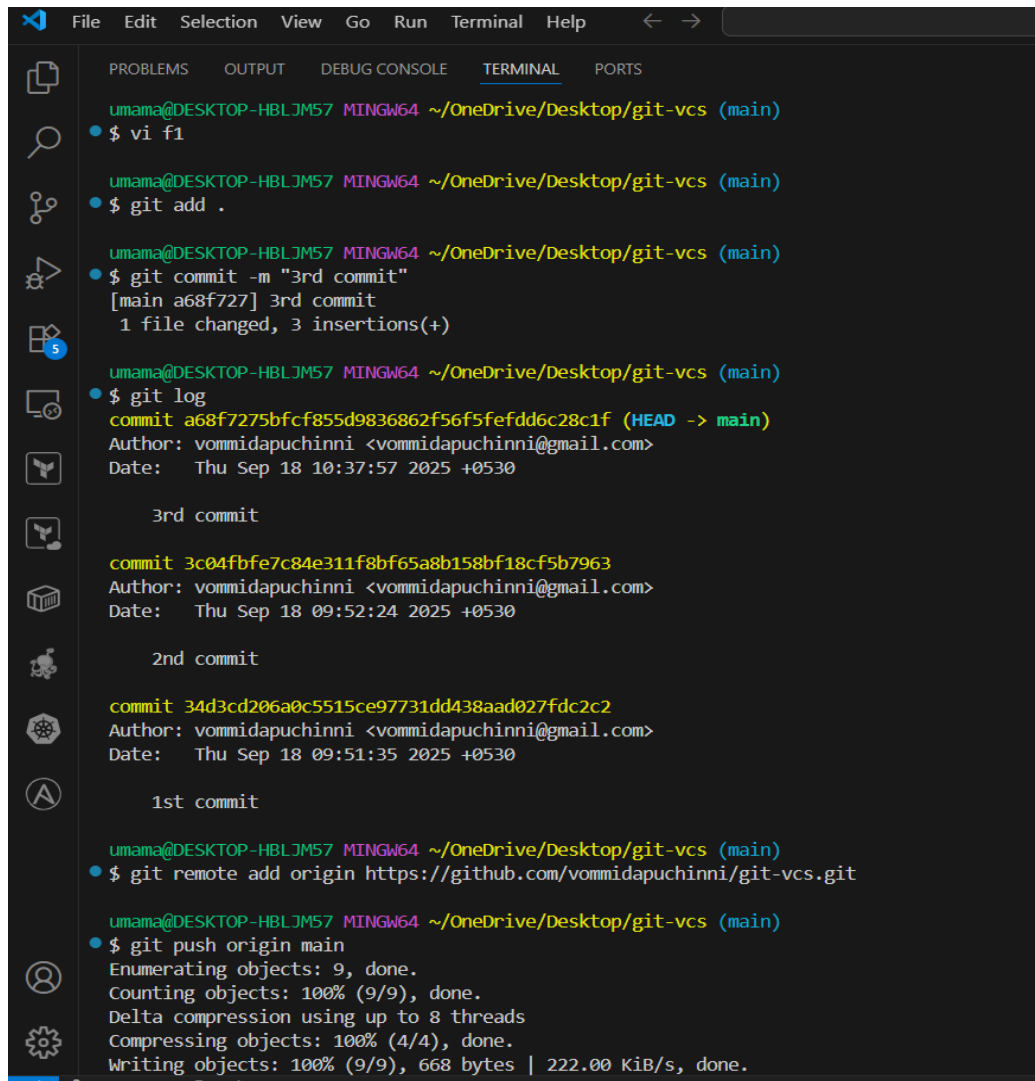
    2nd commit

commit 34d3cd206a0c5515ce97731dd438aad027fdc2c2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date:   Thu Sep 18 09:51:35 2025 +0530

    1st commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ cat f1
aaaaa
bbbbbb
cccccc
AAAAAAA
BBBBBBBB
CCCCCCCC
```

Push those files to remote repo



```
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ vi f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git add .

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git commit -m "3rd commit"
[main a68f727] 3rd commit
1 file changed, 3 insertions(+)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git log
commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:37:57 2025 +0530

    3rd commit

commit 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:52:24 2025 +0530

    2nd commit

commit 34d3cd206a0c5515ce97731dd438aad027fdc2c2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:51:35 2025 +0530

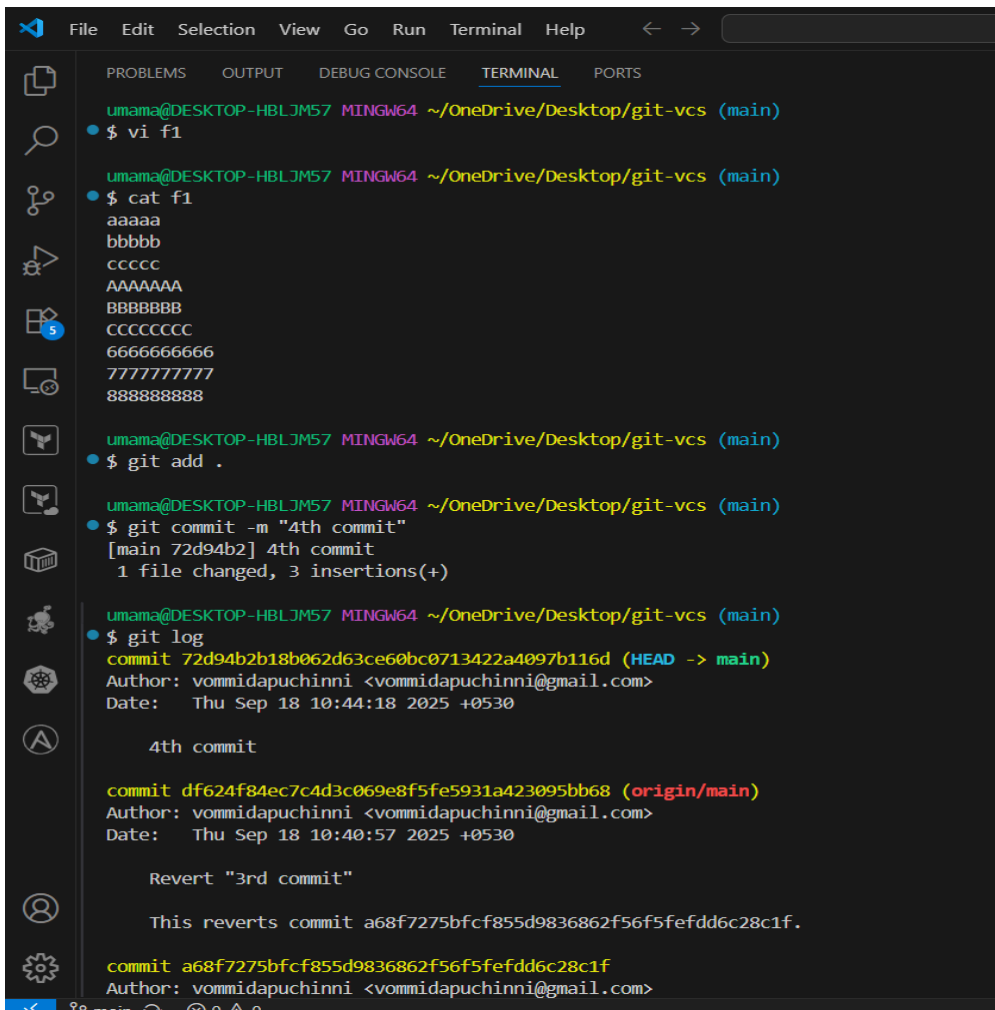
    1st commit

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git remote add origin https://github.com/vommidapuchinni/git-vcs.git

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (9/9), 668 bytes | 222.00 KiB/s, done.
```


7. Git revert: Create a new commit that undoes changes

- In f1 add content and push to remote repo
- Copy 3rd commit id
- `git revert HEAD <paste 3rd commitid>`
- `git log` we see reverted commit and data is deleted in that file



```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ cat f1
aaaaa
bbbbb
cccc
AAAAAAA
BBBBBBB
CCCCCCC
6666666666
7777777777
8888888888

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit -m "4th commit"
[main 72d94b2] 4th commit
1 file changed, 3 insertions(+)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit 72d94b2b18b062d63ce60bc0713422a4097b116d (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:44:18 2025 +0530

    4th commit

commit df624f84ec7c4d3c069e8f5fe5931a423095bb68 (origin/main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:40:57 2025 +0530

    Revert "3rd commit"

    This reverts commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f.

commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f
Author: vommidapuchinni <vommidapuchinni@gmail.com>
```

```
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git revert HEAD a68f7275bfcf855d9836862f56f5fefdd6c28c1f
[main df624f8] Revert "3rd commit"
 1 file changed, 3 deletions(-)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git log
commit df624f84ec7c4d3c069e8f5fe5931a423095bb68 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:40:57 2025 +0530

    Revert "3rd commit"

    This reverts commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f.

commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f (origin/main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:37:57 2025 +0530

    3rd commit

commit 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:52:24 2025 +0530

    2nd commit

commit 34d3cd206a0c5515ce97731dd438aad027fdc2c2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:51:35 2025 +0530

    1st commit


umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ cat f1
aaaaa
bbbbbb
cccccc
AAAAAAA
BBBBBBB
CCCCCCCC

main 0 0 0
```

We see number part is deleted in that file

In remote repo also we can see that commit

Commits		
main	All users	All time
Commits on Sep 18, 2025		
Revert "3rd commit"	vommidapuchinni committed 7 minutes ago	df624f8
3rd commit	vommidapuchinni committed 10 minutes ago	a68f727
2nd commit	vommidapuchinni committed 1 hour ago	3c04fbf
1st commit	vommidapuchinni committed 1 hour ago	34d3cd2

 **8. Git amend:** If you only want to change the last commit message. Not doing the any other new commit keeping or saving data on previous commit

Open the file f1 and add data to it save, add and commit it.

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f1

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ cat f1
aaaaa
bbbbbb
cccccc
AAAAAAA
BBBBBBB
CCCCCCCC
6666666666
7777777777
8888888888

111111111111
222222222222
```

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit --amend -m "4th commit"
[main 324c186] 4th commit
Date: Thu Sep 18 10:44:18 2025 +0530
1 file changed, 3 insertions(+)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git log
commit 324c186ab69180b2ee0c3c866262992b2caf8005 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:44:18 2025 +0530

    4th commit

commit df624f84ec7c4d3c069e8f5fe5931a423095bb68 (origin/main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:40:57 2025 +0530

    Revert "3rd commit"

    This reverts commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f.

commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:37:57 2025 +0530

    3rd commit

commit 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:52:24 2025 +0530

    2nd commit

commit 34d3cd206a0c5515ce97731dd438aad027fdc2c2
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 09:51:35 2025 +0530

    1st commit
```

We want to see latest commits we see git log -2

```

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git log -2
commit 324c186ab69180b2ee0c3c866262992b2caf8005 (HEAD -> main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:44:18 2025 +0530

    4th commit

commit df624f84ec7c4d3c069e8f5fe5931a423095bb68 (origin/main)
Author: vommidapuchinni <vommidapuchinni@gmail.com>
Date: Thu Sep 18 10:40:57 2025 +0530

    Revert "3rd commit"

    This reverts commit a68f7275bfcf855d9836862f56f5fefdd6c28c1f.

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
○ $

```

🔍 9. Difference between to commits copy the commit ids. We see old line in white and new lines in green and deleted lines in red.

```

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ git diff 324c186ab69180b2ee0c3c866262992b2caf8005 3c04fbfe7c84e311f8bf65a8b158bf18cf5b7963
diff --git a/f1 b/f1
index e0fa98c..7008514 100644
--- a/f1
+++ b/f1
@@ -4,6 +4,3 @@ ccccc
AAAAAAA
BBBBBBB
CCCCCCC
-66666666666
-7777777777
-8888888888

```

⊗ 10. .gitignore: Exclude files from being tracked using .gitignore.

```

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi chinni.py

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi .gitignore

```

```

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi hello.java

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
• $ vi .gitignore

```

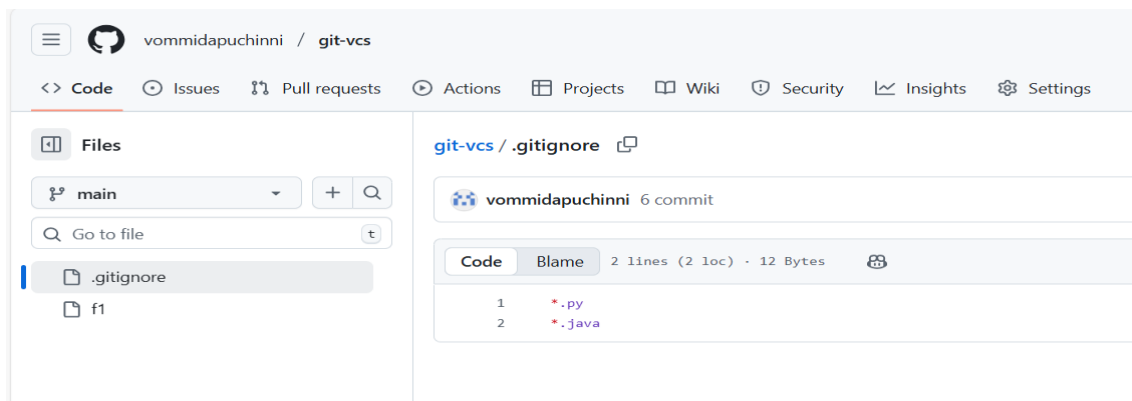
```

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git to

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git commit -m "6 commit"
[main 7ccf126] 6 commit
2 files changed, 5 insertions(+)
create mode 100644 .gitignore

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.

```



11. git pull = git fetch + git merge

```

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git fetch
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 949 bytes | 39.00 KiB/s, done.
From https://github.com/vommidapuchinni/git-vcs
7ccf126..e3e8807  main      -> origin/main

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ ls
chinni.py  f1  hello.java

```

```

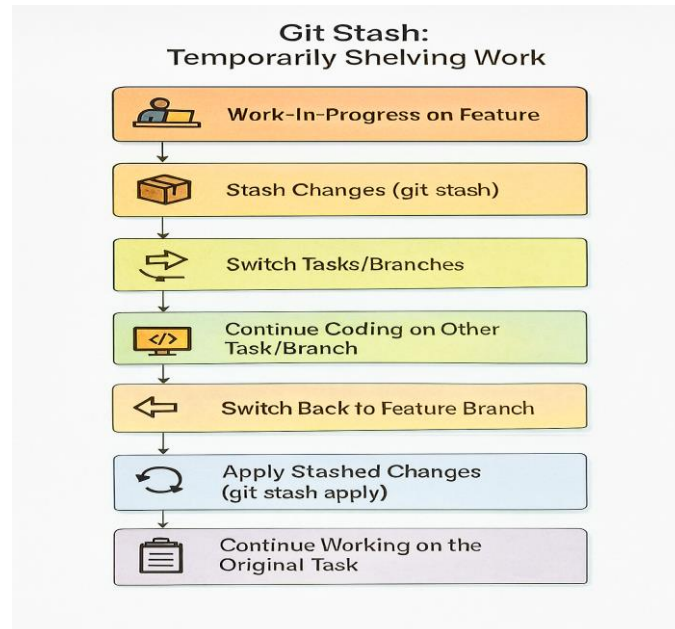
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git merge origin
Updating 7ccf126..e3e8807
Fast-forward
 f2 | 1 +
1 file changed, 1 insertion(+)
create mode 100644 f2

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ ls
chinni.py  f1  f2  hello.java

```

12. Stashing

Definition: Temporarily save uncommitted changes without committing.



- Create a file f3 and add content to it.
- Save it with `git stash save -m "1st stash"`
- `git stash pop` → cut and paste
- `git stash drop` → which stashes are applied those are deleted
- `git stash clear` → clear all stashes
- `git stash list` → list stashes we see

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f3

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ cat f3
111111111111
222222222222222
3333333333333333

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .
warning: in the working copy of 'f3', LF will be replaced by CRLF the next time Git touches it

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git stash save -m "1st stash"
Saved working directory and index state On main: 1st stash

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ cat f3
cat: f3: No such file or directory
```

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ git stash list
stash@{0}: On main: 1st stash

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ git stash save
No local changes to save

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ git stash apply
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   f3

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ ls
chinni.py  f1  f2  f3  hello.java

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
○ $
```

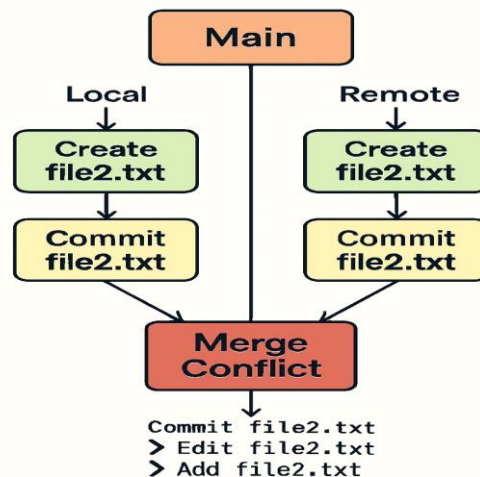
```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ git stash pop
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   f3

Dropped refs/stash@{0} (d593039e5c320d35618360fadaedf457a52252da)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ git stash drop
Dropped refs/stash@{0} (645e6ad494a19967c870a22c58ea46ccc3b49f98)

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
● $ git stash clear
```

✂ **13. Merge conflict:** A merge conflict occurs when Git cannot automatically merge changes because two branches modified the same part of a file differently. The developer must manually resolve the conflict before completing the merge.



- Create file in repo and add some content and commit changes
- Now in local directory create file with same name created in repo.
- Now git add that file and commit it.
- Now when we push we get merge conflict.

```
umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git stash clear

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ vi f4

umama@DESKTOP-HBLJM57 MINGW64 ~/OneDrive/Desktop/git-vcs (main)
$ git add .
warning: in the working copy of 'f4', LF will be replaced by CRLF the next time Git touches it
```

When we get merge conflict, we first pull remote repo after that, Open the file and remove unwanted data.


```

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$ git push origin main
To https://github.com/vommidapuchinni/git-vcs.git
! [rejected]        main -> main (non-fast-forward)
error: failed to push some refs to 'https://github.com/vommidapuchinni/git-vcs.git'
hint: Updates were rejected because the tip of your current branch is behind
hint: its remote counterpart. If you want to integrate the remote changes,
hint: use 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$ git pull origin main
From https://github.com/vommidapuchinni/git-vcs
* branch            main      -> FETCH_HEAD
Auto-merging f4
CONFLICT (add/add): Merge conflict in f4
Automatic merge failed; fix conflicts and then commit the result.

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|MERGING)
$ cat f4
<<<<<<< HEAD
11111111111111111111
=====
all ones
>>>>>> 598bcb4d6bf417046a2b058b6233fec57f9d92e0

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|MERGING)
$ vi f4

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|MERGING)
$ vi f4

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|MERGING)
$ git add .

```

After that add that file and commit it.

Now push the file

```

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|MERGING)
$ cat f4
11111111111111111111

all ones

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|MERGING)
$ git commit -m "commit10"
[main 687761f] commit10

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$ git push origin main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (11/11), 873 bytes | 174.00 KiB/s, done.
Total 11 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/vommidapuchinni/git-vcs.git
598bcb4..687761f  main -> main

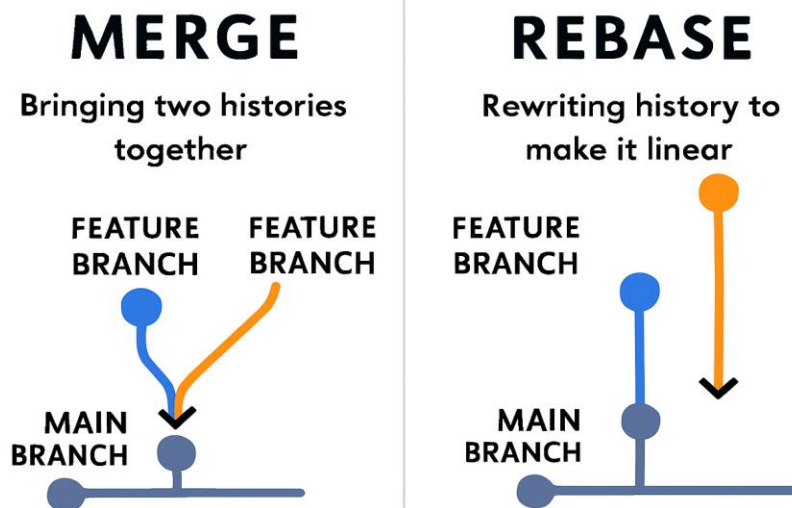
```

□ **14. Rebase** = move or replay commits from one branch on top of another.

Instead of merging, which creates a merge commit, rebase rewrites history so it looks like your work started from the latest commit in the target branch.

In simple terms:

- Merge = “combine both histories with all bumps.”
- Rebase = “lift my commits and put them on top of the other branch (clean history).”



`git log --oneline` → get all logs in one line

Since I want to rewrite the last 2 commits (commit10 and commit): `git rebase -i HEAD~2`

```

pick 6dcb927 # commit
squash 598bcb4 # commit9

# Rebase 7461bf5..687761f onto 7461bf5 (2 commands)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous
#   commit's log message, unless -C is used, in which case
#   keep only this commit's message; -c is same as -C but
#   opens the editor
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
#   create a merge commit using the original merge commit's
#   message (or the oneline, if no original merge commit was
#   specified); use -c <commit> to reword the commit message
# u, update-ref <ref> = track a placeholder for the <ref> to be updated
#   to this position in the new commits. The <ref> is
#   updated at the end of the rebase
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
~
~

```

This means “combine commit10 into commit”

- If you want to drop commit9 completely: pick 6dcb927 # commit && drop 598bcb4 # commit9
- If you want to reword the message of commit9: pick 6dcb927 # commit && reword 598bcb4 # commit9

```

Auto-merging f4
or your editor to close the file...Rebasing (2/2)
CONFLICT (add/add): Merge conflict in f4
error: could not apply 598bcb4... commit9
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git r
Could not apply 598bcb4... # commit9

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|REBASE 2/2)
$ git status
interactive rebase in progress; onto 7461bf5
Last commands done (2 commands done):
  pick 6dcb927 # commit
  squash 598bcb4 # commit9
No commands remaining.
You are currently rebasing branch 'main' on '7461bf5'.
  (fix conflicts and then run "git rebase --continue")
  (use "git rebase --skip" to skip this patch)
  (use "git rebase --abort" to check out the original branch)

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
    both added:      f4

no changes added to commit (use "git add" and/or "git commit -a")

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|REBASE 2/2)
$ cat f4
<<<<<<< HEAD
11111111111111111111
=====
all ones
>>>>>> 598bcb4 (commit9)

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main|REBASE 2/2)
$

```

Edit file and push it.

```

[detached HEAD 09908da] commit
Date: Thu Sep 18 11:20:25 2025 +0530
4 files changed, 7 insertions(+)
create mode 100644 .f3.swp
create mode 100644 .f4.swp
create mode 100644 f4
Successfully rebased and updated refs/heads/main.

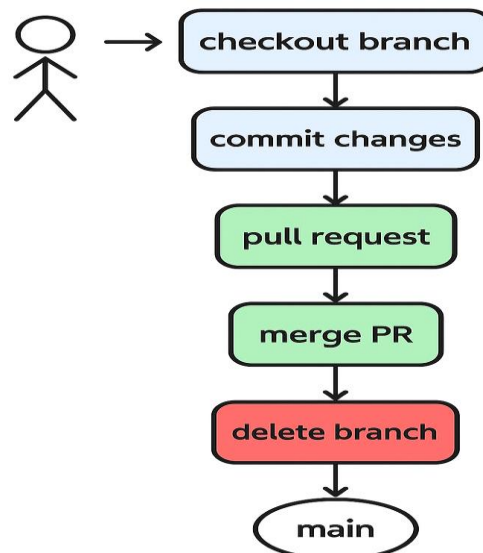
umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$ git push origin main --force
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 445 bytes | 148.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/vommidapuchinni/git-vcs.git
+ 687761f...09908da main -> main (forced update)

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$

```

📌 **15. A Pull Request** is a feature in Git platforms like GitHub, GitLab, or Bitbucket that lets you propose changes made in one branch to be merged into another branch (usually main or develop).

- `git checkout -b feature-branch`
- `echo "Some change" >> test.txt`
- `git add test.txt`
- `git commit -m "Add test file for PR"`
- `git push origin feature-branch`



```
umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$ git push origin feature-branch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:   https://github.com/vommidapuchinni/git-vcs/pull/new/feature-branch
remote:
To https://github.com/vommidapuchinni/git-vcs.git
 * [new branch]       feature-branch -> feature-branch
```

```

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$ echo "Some change" >> test.txt

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$ git add test.txt
warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next time Git touches it

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$ git commit -m "Add test file for PR"
[feature-branch d44d54d] Add test file for PR
1 file changed, 1 insertion(+)
create mode 100644 test.txt

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$ git push origin feature-branch
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 287 bytes | 71.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/vommidapuchinni/git-vcs.git
09908da..d44d54d feature-branch -> feature-branch

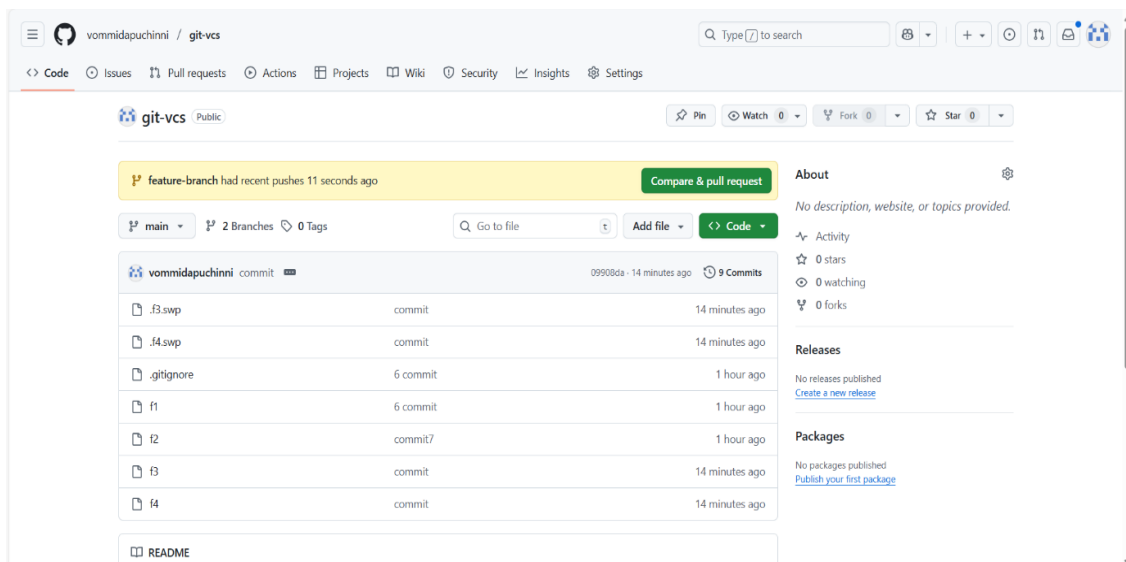
umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$

```

Go back to GitHub → Pull Requests → New pull request.

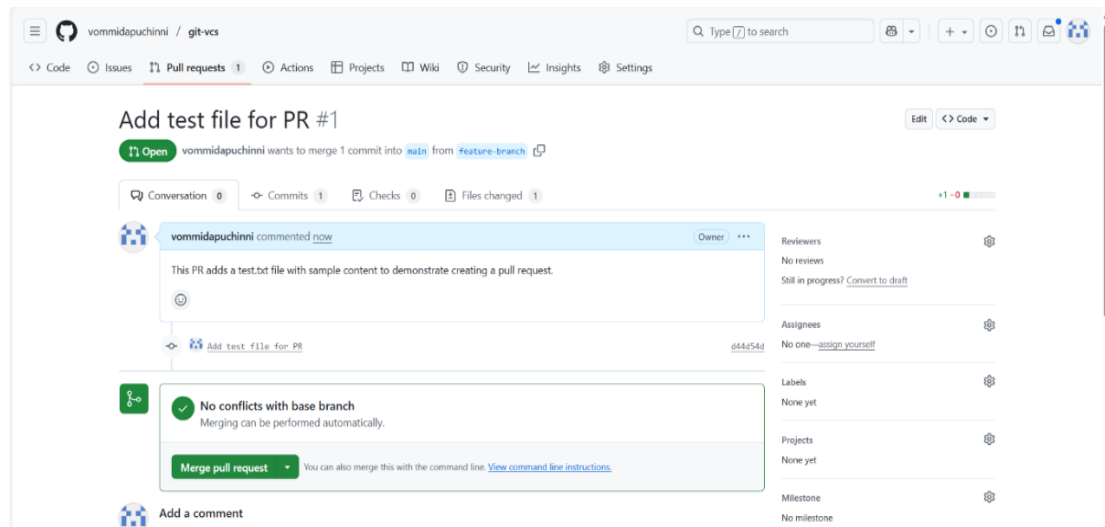
- Now you'll see the changes from feature-branch compared to main.

Create the PR as usual.

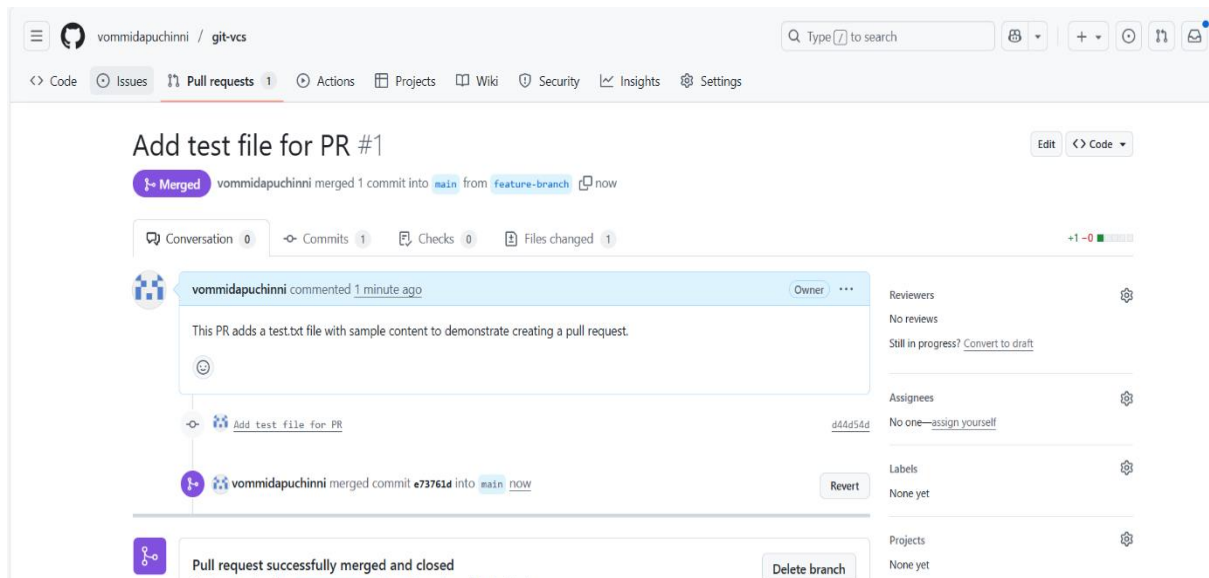


Steps to create the PR

- Title: Add test file for PR (or any descriptive title)
- Description: Explain what this PR does.
- Click Create Pull Request.



Click on merge pull request both branches will be merged

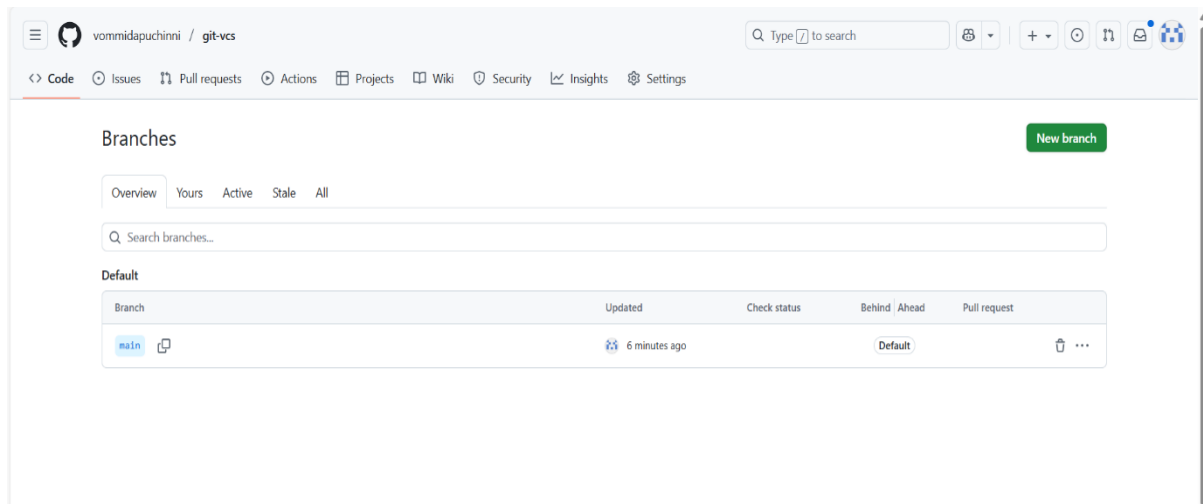


Delete branch in repo and local we use git push origin --delete branchname

```
umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (feature-branch)
$ git checkout main
Switched to branch 'main'

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
$ git push origin --delete feature-branch
To https://github.com/vommidapuchinni/git-vcs.git
- [deleted]          feature-branch

umama@DESKTOP-HBLJM57 MINGW64 /d/git-vcs (main)
```



□ 16. Advanced topics:

- □ Hooks: Automate actions on commits (.git/hooks/pre-commit)
- □ Bisect: Find bad commit: git bisect start
- □ Submodules: Repo inside a repo: git submodule add <url>
- 🏠 Worktree: Multiple working directories: git worktree add ../feature feature-branch